Multi-threaded parallel simulation of non-local non-linear problems in ultrashort laser pulse propagation in the presence of plasma

Mandana Baregheh[a] , Vladimir Mezentsev[a], Holger Schmitz[b]

[a]Photonics Research Group, Aston University, Birmingham, B4 7ET, UK
[b]Plasma Physics Group, Imperial College London, South Kensington Campus, London SW7 2AZ, UK

ABSTRACT

We describe a parallel multi-threaded approach for high performance modelling of wide class of phenomena in ultrafast nonlinear optics. Specific implementation has been performed using the highly parallel capabilities of a programmable graphics processor.

Keywords: Femtosecond phenomena, Non-linear Schrödinger equation, Parallel numerical simulations

1. INTRODUCTION

Nonlinear wave equation in the form of the Generalised Non-Linear Schrödinger Equation (GNLSE) is a generic mathematical model describing narrow bandwidth wave propagation in envelope approximation. In this paper we consider a parallel numerical solution for a specific model which describes femtosecond laser pulse propagation in transparent media. In this model, GNLSE is coupled to the Drude model of plasma resulting from multi-photon and avalanche ionisation processes [1, 2, 3]. However our approach can be extended to similar models.

We compare performance of the multithreaded parallel code implemented for Nvidia Graphics Processing Units (GPU) using CUDA programming interface [4] with a serial CPU version similar to that described in [1, 2]. Coupling between the GNLSE and the Drude model of plasma results in impossible straightforward parallel implementation of such model due to non-locality. Hence, efficient use of GPU's parallel resources requires radical revision of the current pipelined method presented in [1, 2].

Parallel solution of NLS like systems have been discussed previously by Zoldi in [5] for large scale problems, where parallelization of the Fast Fourier Transform (FFT) is addressed. Traditionally, numerical solution of GNLSE-type problems i.e. partial differential equations comprising *linear* differential operator and *local* nonlinear terms is addressed by operator splitting techniques whereby a problem specific succession of linear and nonlinear operators approximate the original equation [6, 7]. Solution of linear problem often referred to as a *linear step* is usually efficiently performed using spectral methods which often have efficient parallel numerical implementations such as FFT. Efficiency of spectral methods results from reducing a solution discretised linear partial differential equation as a matrix inversion problem into a set of trivial uncoupled equations in corresponding spectral domain. Similarly, nonlinear problem, or *nonlinear step*, is usually already local i.e. already represents a set of uncoupled equations. Thus operator splitting makes possible efficient parallel implementations for equations with local nonlinearities as shown e.g. in [5].

In this paper we consider a more complicated model which appears in describing ultrashort laser pulse propagation in transparent media (see e.g. [8]). Typical applications of such phenomenon are femtosecond (fs) laser inscription for microfabrication and micro-machining [3]. A fundamental difference of relevant mathematical models to NLSE equations is in the nature of nonlinear terms which are nonlocal in time due to peculiar interaction of paser light with plasma. This paper addresses an elimination of numerical bottleneck associated with nonlocal nature of the nonlinear terms in the wave equation. A conventional operator splitting technique

Further author information: (Send correspondence to M. Baregheh)
E-mail: bareghem@aston.ac.uk

Nonlinear Optics and Applications V, edited by Mario Bertolotti, Proc. of SPIE Vol. 8071
807115 · © 2011 SPIE · CCC code: 0277-786X/11/$18 · doi: 10.1117/12.886822

is exploited to effitiently deal with the linear operator while special attention is paid to constructing an efficient parallel method of integration of the nonlinear operator. We adopt the split-step Fourier method for solution of GNLSE. Since the typical problem size of $10^4 \times 10^4$ fits to the GPU memory well developed existing numerical implementations are used such as an optimised off-the shelf FFT.

2. THEORETICAL MODEL

High intensity laser field used in the above mentioned applications results in complex processes of material ionisation and evolution of plasma which comprises cascaded multi-photon absorption, plasma absorption, and defocusing. As a result the propagation appears to be extremely complex affected by a combination of multiple nonlinear effects.

In this paper, we consider a mathematical model being an extended non-local version of NLSE which is derived from coupling between GNLSE and the particle balance equation for plasma resulting from multi-photon and avalanche ionisation processes. The creation of plasma results from multi-photon absorption leading to resistive absorption of laser light and its defocusing. These effects together with Kerr effect, diffraction and dispersion make the pulse propagation a very complex phenomenon developing over a range of scales in space and time [2].

We illustrate our approach for the following model describing nonlinear propagation of intense ultrashort laser pulse in transparent dielectric media. We use a set of equations equivalent to that used by Feng et al [9]. An extensive discussion of more advanced models can be found in a recent review [8]. Note that our aproach can be exploited for more complicated forms of the wave equations of NLSE type.

The first equation is the NLSE describing an envelope amplitude u of the laser wave coupled to an equation describing concentration of plasma carriers ρ along the z-axis:

$$\partial_z u - i\kappa\Delta_\perp u - i\sigma|u|^2 u = -id\partial_{tt}u - \gamma(1 + i\omega\tau)\rho u - \mu|u|^{2(K-1)}u \; ; \tag{1}$$

$$\partial_t \rho = \nu\rho|u|^2 + \left(\frac{|u|}{u_0}\right)^{2K} \; . \tag{2}$$

The notation used in this paper and parameters are described in our earlier papers [1], [2]. Here u stands for the envelope amplitude of electric field with angular frequency ω and the wave vector k:

$$E = u(r_\perp, t, z)e^{i(\omega t - kz)} \; .$$

The terms in the left hand-side of Eq.(1) describe the effects of beam diffraction and Kerr self-refraction. These terms alone form the well known NLSE. Kerr effect in 2D and 3D geometries leads to a singular catastrophic self-focusing which is ultimately arrested by effects of dispersion [10], plasma absorption and defocusing, and multi-photon absorption (MPA) [2]. The first term in the right hand-side of equation (1) describes dispersion. The second term on the right hand side ($\gamma(1 + i\omega\tau)\rho u$) comprises effects of plasma absorption and defocusing and the final term in this equation describes MPA.

Equation (2) describes evolution of plasma density ρ. The first term in the right hand side describes avalanche ionization and the second term describes multi-photon ionization.

Extended NLSE in the form of Eqs.(1,2) is a system of nonlinear partial differential equations which can not be solved analytically. Two main numerical approaches have been used for solution of NLSE known as finite-difference method and pseudo-spectral methods. The coupling between the equations makes NLSE non-local which results in impossible straightforward parallel implementation typical for explicit finite difference methods. The pseudospectral methods are proven to be faster and in that category spilt-step Fourier method is proven to be the most efficient [6, 7].

The splitting operator method is used here similar to that in [1] to reduce extended NLSE into a succession of linear and nonlinear steps. The split-step method derives an approximate discretised solution by considering effects of linearity and nonlinearity independently by dividing the problem into two sequential steps where in the first step nonlinearity acts alone and the second step is for independent linear effect as shown below[7]:

$$\partial_z u = Lu = -i\kappa\Delta_\perp u + id\partial_{tt} u = 0 \; ; \tag{3}$$

$$\partial_z u = Nu = -i\gamma(1 + i\omega\tau)\rho u - \sigma|u|^2 u - i\mu|u|^{2(K-1)} u \; . \tag{4}$$

Here L and N schematically denote the linear and nonlinear operators correspondingly. It can be shown that consecutive application of linear and nonlinear operators after finite increment of evolution variable z approximates the original system of differential equations Eqs.(1,2):

$$u(z + \Delta z) = \exp\left(\frac{N}{2}\Delta z + L\Delta z + \frac{N}{2}\Delta z\right) u(z) + O(\Delta z^3) \; . \tag{5}$$

2.1 Initial Condition

We used radially symmetric initial conditions having Gaussian shape in radial and temporal dimensions presented in [2], [9] :

$$u(z = 0, r_\perp, t) = \sqrt{\frac{2P_{in}}{\pi r_0^2}} \exp\left(-\frac{r^2}{r_0^2} - \frac{ikr^2}{2f} - \frac{t^2}{t_p^2}\right) \; , \tag{6}$$

where r_0 stands for the waist of the incident beam, t_p is the pulsewidth, f is the focal length of the lens and P_{in} is the peak input power [2], [9]. The initial pulse is stored in a complex matrix $u(m, n)$ which is a discretisation of continuous field $u(r, t)$. To boost performance,this matrix is physically stored as a 1D array of size $N_r \times N_t$ where N_r and N_t stand for the number of grid points in radial and time direction accordingly. Every row in this matrix contains $u(t)$ for a fixed radial position. The array is periodically reorganised using parallel routines to perform matrix transposition in order to provide continuous storage for coordinate-wise parallel operations such as Fourier transform and integration of the radial operator using Cranck-Nicholson scheme. The numerical step in evolution variable z was varied to ensure a proper approximation of discrete numerical scheme.

3. GRAPHICS PROCESSING UNITS AND CUDA

Graphics Processing Units are specialised processors that are designed for highly parallel computations needed for graphics rendering where more arithmetic units are devoted to data processing rather than data cache and flow control. Their highly parallel capabilities makes them suitable for complex algorithms, However they require low level programming and fine adjustment of the algorithms in order to ensure concurrent independent access to data [4].

To design an efficient and optimised numerical model, we have chosen to use the highly parallel capabilities of a programmable graphics processor from Nvidia's products through the CUDA (Compute Unified Device Architecture) interface. CUDA links the application software and hardware required for extremely parallel computation systems [4]. It is essentially a commodity solution that enables low cost parallel programming possible via flexible and easily adaptable extension of the C language. The C language interface provided by CUDA is far less complicated compared to other graphics API like OpenGL.

In order to efficiently exploit CUDA's parallel resources, radical revision is required of the traditional split-step procedures routinely used for integration of NLSE type models. Parallel execution of the code by multiple threads increases the computation throughput and achieves more accurate results whilst reducing the processing time [11].

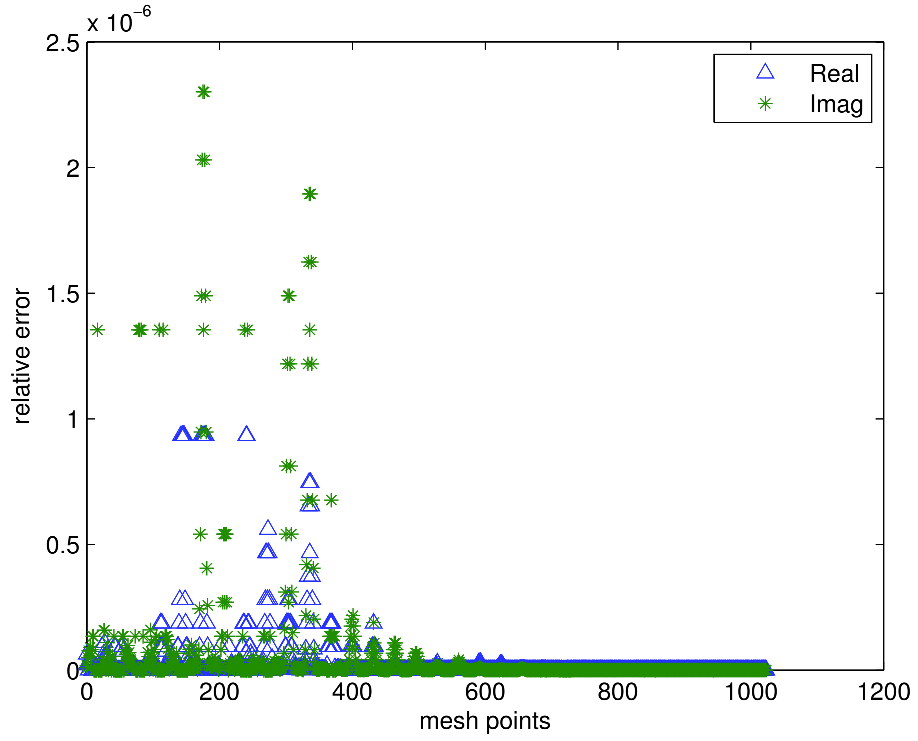

Figure 1. Difference between the real and imaginary part of the result produced by CPU and GPU for calculation of the linear term.

4. PARALLELISATION

As mentioned earlier, the splitting operator method is used to simplify the non-local non-linear NLSE to reduce it into a succession of linear and nonlinear steps.

To solve the linear term (Eq. 3), Fast Fourier Transform is applied in the time domain which results in a linear radial problem for each temporal frequency mode as shown below:

$$\partial_z \tilde{u} = -i\kappa\Delta_\perp \tilde{u} + id\omega^2 \tilde{u} \ , \quad \text{where} \quad \tilde{\mathrm{u}} = \mathrm{F} \cdot \mathrm{u} \ . \tag{7}$$

FFT is applied using `cufft`, CUDA's own implementation of FFT on batches of size N_t for every row of data [12]. To access continuous chunks of memory more efficiently, parallel transposition is implemented on every row of data. Subsequently an implicit finite difference solution is used for Eq.(6) which uses a tridiagonal solver for Cranck-Nicholson discretisation in evolution variable z. Figure 1 plots the difference between the result obtained from calculation of the linear term on CPU(using a serial version similar to that described in [1], [2]) and GPU(on Nvidia's CUDA) for problem of size 32^2. Here relative error is calculated by $|u_{cpu} - u_{gpu}|/\max|u_{cpu}|$, where indices *cpu* and *gpu* represent results obtained from the calculations on CPU and GPU accordingly. It can be seen that the relative error is in the acceptable range of 10^{-6} for single floating point precision.

While `cufft` and concurrent execution of tridiagonal solver provide a satisfactory scalable parallel solution for numerical integration of the linear operator, scalable implementation of nonlinear step is tricky since it involves a convoluted nonlocal operator.

In the non-linear term, non-local dependence of ρ on u results in global coupling. Eq.(2) is usually solved using one of the methods for numerical integration for first order ordinary differential equation. It is essentially serial procedure. On the other hand, Eq.(2) for a given intensity $|u|^2$ is a linear equation for ρ which can be

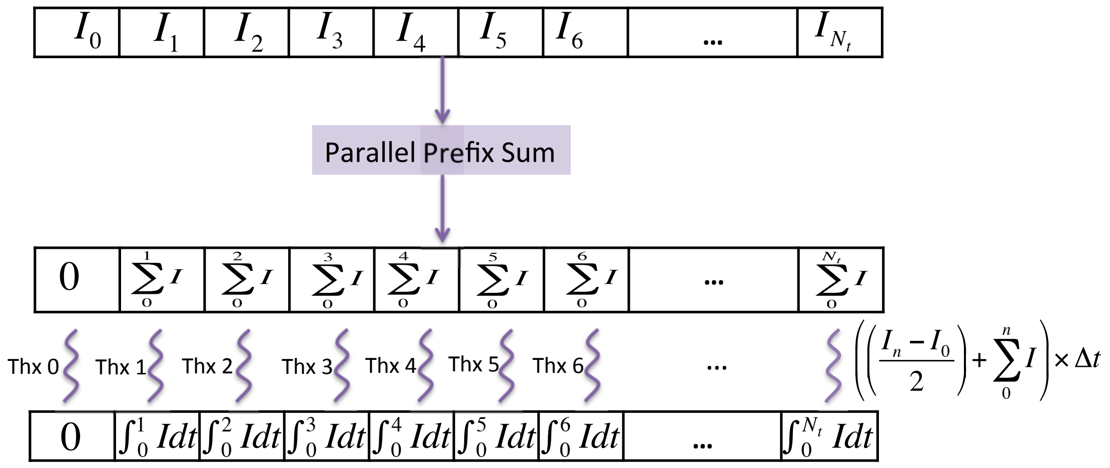

Figure 2. Parallel calculation of $\int Idt$ required for Eq. 8.

analytically integrated to yield a solution. Since the differential solution is serial and integration can be made parallel easier, integration is used to parallelise the solution by splitting the problem as an integral equation in time domain. The solution of Eq.(2) for a given intensity $I(t)$ is:

$$\rho(t) = \exp(\varphi(t)) \int_{-\infty}^{t} I^K \exp(-\varphi(t)dt \ , \tag{8}$$

where

$$\varphi(t) = \int_{-\infty}^{t} Idt \ .$$

Note that, this solution is provided for initial condition for Eq.(2).

In our numerical implementation of Eq.(8), we delegate partial numerical integration using trapezoidal rule to concurrently running threads as illustrated in Figure 2. Here the parallel prefix sum implementation described in [13] provided by Nvidia corporation is used to perform scan sum of the data using a balanced tree algorithmic pattern.

Figure 3, plots the maximum relative error calculated from comparison of the results obtained from CPU and GPU for different number of mesh points starting from 32×32. Both CPU and GPU results are obtained from completion of one step combining both the linear and nonlinear implementation of NLSE(Eq. 5). As illustrated in this figure, as the problem size increases, the maximum error becomes very small in the range of 10^-6 which is limited by arithmetic precision. Since the implementation of this code differs in CPU and GPU due to peculiar parallelisation of the GPU code, this proves that both method converge with sufficiently high accuracy at finer resolutions as illustrated in Fig 3.

For the experimental setup, we have used an Nvidia Tesla C1060 with Nvidia driver 195.36 and CUDA version 3.0. The C1060 has 30 streaming multiprocessor(SM), where each multiprocessor is comprised of 8 streaming processor(SP), providing us with a total of 240 SPs. A 16KB shared memory is assigned to each SM providing a fast low-latency data cache. Also there is 4GB of device memory available on the C1060.

The performance comparison has been done between a CPU serial code similar to that described in [1], [2] and a GPU parallel code. The execution time of CPU and GPU implementation is compared in Figure 4, which is calculated by taking the wall clock time on both CPU and GPU including the initial set-up and the data transfer time from CPU's host memory to GPU's global memory. As the problem size increases the GPU comuting speed

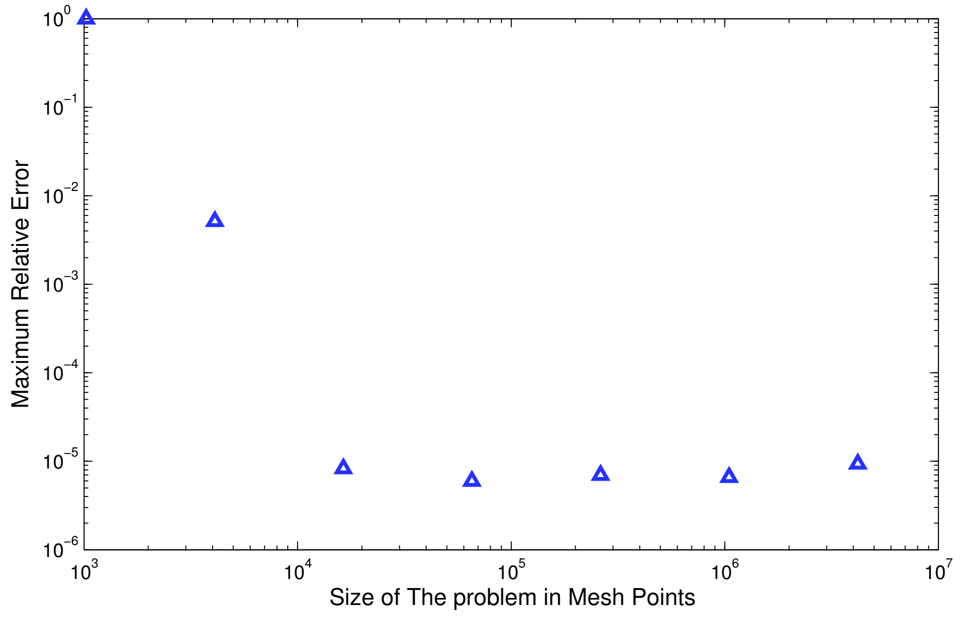

Figure 3. Relative error derived from comparison of results produced by CPU and GPU for different number of mesh points starting from $N_r \times N_t = 32 \times 32$ up to $N_r \times N_t = 2048 \times 2048$.

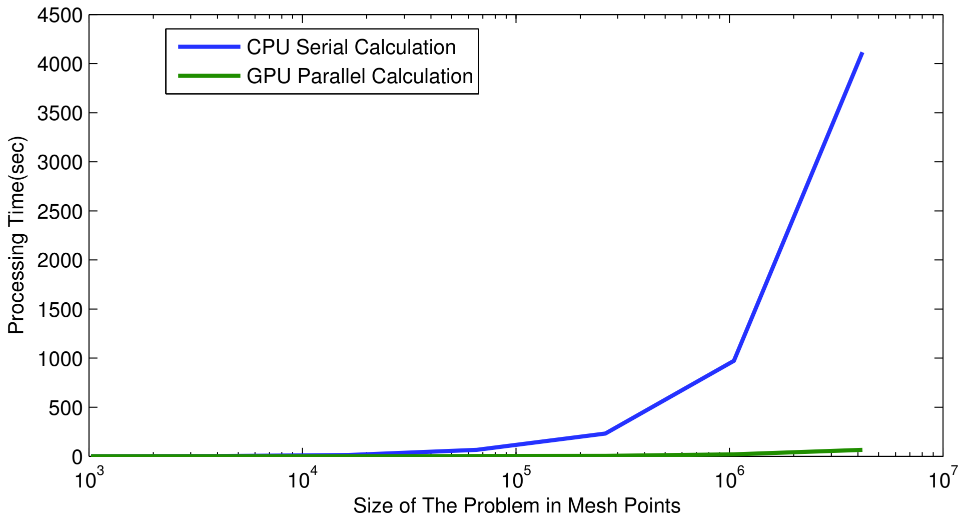

Figure 4. CPU vs GPU comparison of performance time for calculation of both linear and nonlinear term

is notably increased providing us with a 63x speed-up. Here the speed-up is calculated by dividing the execution time on CPU by the required execution time on GPU.

5. CONCLUSION

We have managed to successfully parallelise solution of non-local extended NLSE through CUDA interface using multithreaded graphical hardware. We compared performance of the described above parallel code implemented for Nvidia Graphics Processing Units using CUDA programming interface with a serial CPU version used in

[1,2]. The result of this comparison showed us a satisfactory 10^{-6} difference between GPU and CPU results for single floating point precision. As for performance time, we have up to $\times 60$ speedup on a Tesla C1060 Graphics card compared to a Intel Core i7.

REFERENCES

[1] Mezentsev, V., Petrovic, J., Dreher, J., and Grauer, R., "Adaptive modeling of the femtosecond inscription in silica," *Proc. SPIE, Laser-based Micropackaging* **6107**, 241–250 (2006).

[2] Mezentsev, V., Petrovic, J., Dubov, M., Bennion, I., Dreher, J., Schmitz, H., and Grauer, R., "Femtosecond laser microfabrication of subwavelength structures in photonics," *Proc. SPIE, Laser-based Micropackaging* **6459**, 64590B (2007).

[3] Gattass, R. R. and Mazur, E., "Femtosecond laser micromachining in transparent materials," *Nature Photonics* **2**, 219–225 (2008).

[4] "Nvidia cuda programming guide." `http://developer.nvidia.com/object/gpu_programming_guide.html` (2010).

[5] Zoldi, S., Ruban, V., Zenchuk, A., and Burtsev, S., "Parallel implementations of the split-step fourier method for solving nonlinear schroedinger systems," *SIAM News* , 8–9 (1999).

[6] Taha, T. R. and Ablowitz, M. J., "Analytical and numerical aspects of certain nonlinear evolution equations. ll. numerical, nonlinear schroedinger equation," *Journal Of Computational Physics* **55**, 203 (1984).

[7] Agrawal, G. P., [*Nonlinear Fiber Optics*], Academic Press, Burlington, MA ; London (1951).

[8] Bergé, L., Skupin, S., Nuter, R., Kasparian, J., and Wolf, J.-P., "Ultrashort filaments of light in weakly-ionized, optically-transparent media," *Reports on Progress in Physics* **70**, 1633–1713 (2007).

[9] Feng, Q., Moloney, J. V., Newell, A., Wright, E., Cook, K., Kennedy, P., Hammer, D., Rockwell, B., and Thompson, C., "Theory and simulation on the threshold of water breakdown induced by focused ultrashort laser pulses," *IEEE Journal of Quantum Electronics* **33, No. 2**, 127–137 (1997).

[10] Germaschewski, K., Grauer, R., Bergé, L., Mezentsev, V. K., and Rasmussen, J. J., "Splittings, coalescence, bunch and snake patterns in the 3d nonlinear schroedinger equation with anisotropic dispersion," *Physica D* **151**, 175–198 (2001).

[11] Graham, S. L., Snir, M., and Patterson, S. A., [*Getting Up to Speed: The Future of Supercomputing*], National Academies Presss, Washington, D.C. (2004).

[12] "Cuda cufft library." `http://www.nvidia.com/object/cuda_documentation_stage.html` (2007).

[13] S. Sengupta, M. Harris, Y. Z. and Owens, J. D., "Scan primitives for gpu computing," *Proceedings of Graphics Hardware* (2007).